Big Data Computing - Top 40 Music Sentiment Analysis

Christopher Ravenscroft and Harry Graves

December 2023

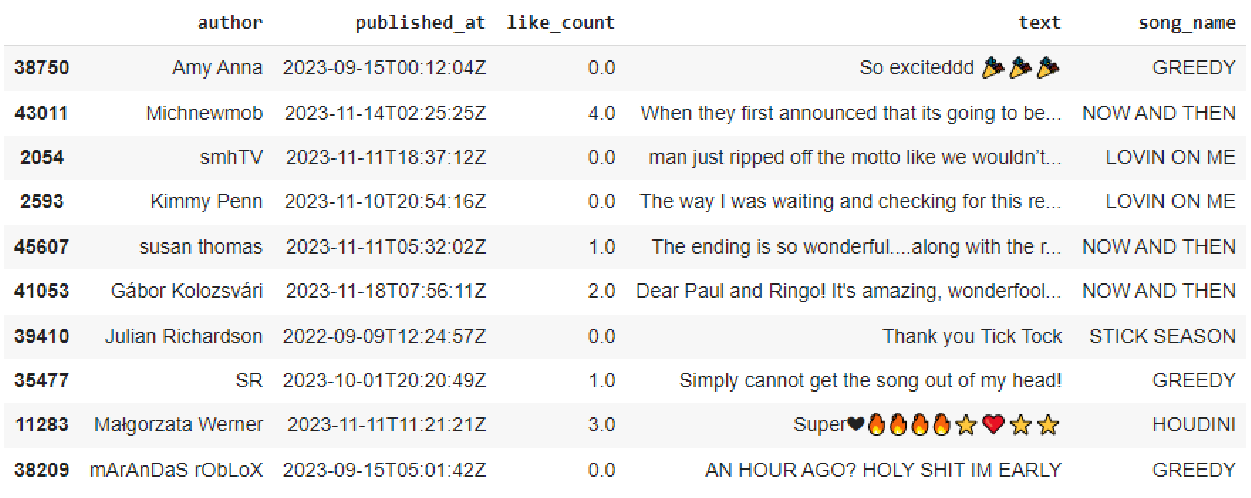
Introduction

In this report we will be attempting to determine a relationship between the online perception of a given song, and it's position in the UK top 40 charts[3]. To achieve this, we have the following aims. First, we wish to obtain a large volume of data, using which we will train a sentiment analysis model. We then wish to improve this model, and provide a comprehensive review of the batch data, with a dashboard with various visualizations to help, which we will use to gain insight on the data. Finally, we wish to apply our model to new data in real time, and improve the dashboard to be able to do this. Through this, we hope to gain understanding as to whether there is any relation to the charts ranking by the positive online perceptions of the song.

We would like to acknowledge the use of AI services in our project. Specifically, Chat-GPT was used to help troubleshoot problems in our code, and improve readability of code. We also obtained some solutions for scraping data from users on github, analyticswithadam [1] and bocchilorenzo [2].

For our data, we decided to obtain batch data from Youtube Comments. Youtube is a leading service provider for viewing and reviewing music, with videos containing thousands of comments, especially those of highly streamed top 40 songs. Along with this, Youtube has a dedicated subsection, Youtube Music, designed for users to listen to music. Hence it is a suitable to obtain a huge volume of comments. Similarly, it made it simple to ensure comments were relevant. Song titles can be commonly used words or phrases, hence searching for related comments through over social medias often provided lots of junk. Finally, the Youtube API made it extremely simple and free to obtain these comments. Hence with the ease of obtaining relevant comments and huge volume, it was an obvious decision to use Youtube comments.

Figure 1 shows a randomly taken sample of 10 rows of our batch data set. An 'author' column a categorical variable which would hold the username of the commenter, 'published_at' a categorical variable holding the time at which the comment was posted, 'like_count' a numerical variable the amount of likes the comment has and 'text' a categorical variable, with the content of the comment itself. Even though 'published_at' is a categorical variable, it is in the syntax form of the date data type, YYYY-MM-DD with extra information on the minute of the post in the form HH-MM-SS added onto the end, conforming to generalised data analysis



	author	published_at	like_count	text	song_name
38750	Amy Anna	2023-09-15T00:12:04Z	0.0	So exciteddd 🎉🎉🎉	GREEDY
43011	Michnewmob	2023-11-14T02:25:25Z	4.0	When they first announced that its going to be...	NOW AND THEN
2054	smhTV	2023-11-11T18:37:12Z	0.0	man just ripped off the motto like we wouldn't...	LOVIN ON ME
2593	Kimmy Penn	2023-11-10T20:54:16Z	0.0	The way I was waiting and checking for this re...	LOVIN ON ME
45607	susan thomas	2023-11-11T05:32:02Z	1.0	The ending is so wonderful....along with the r...	NOW AND THEN
41053	Gábor Kolozsvári	2023-11-18T07:56:11Z	2.0	Dear Paul and Ringo! It's amazing, wonderfool...	NOW AND THEN
39410	Julian Richardson	2022-09-09T12:24:57Z	0.0	Thank you Tick Tock	STICK SEASON
35477	SR	2023-10-01T20:20:49Z	1.0	Simply cannot get the song out of my head!	GREEDY
11283	Małgorzata Werner	2023-11-11T11:21:21Z	3.0	Super🖤🔥🔥🔥🔥⭐❤️⭐⭐	HOUDINI
38209	mArAnDaS rObLoX	2023-09-15T05:01:42Z	0.0	AN HOUR AGO? HOLY SHIT IM EARLY	GREEDY

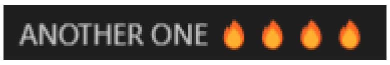
Figure 1: Random Sample of the Pre-Processed Batch Data.

syntax. After attempting to scrape all of the comments for top 40, we reached a limit built into the Youtube API used to scrape comments, this meant that the algorithm stopped at the top 25 songs, scraping just over 500,000 comments. This did not affect the analysis has this gave more than enough data to train a good predictive model and did not affect our goals.

Preprocessing

We initially classified the comments using the popular python package TextBlob, which allows natural language processing on text. TextBlob returns two values when applied, polarity and subjectivity. Subjectivity is defined as a float value from 0 to 1, and describes how factual the statement is, 0 being completely opinion and 1 being completely fact. Polarity however is the value we are interested in; it is defined as a float value ranging from -1 to 1 and describes the sentiment of the text, -1 being very negative and 1 is very positive[4].We will then use this value to classify each comment into either negative, neutral or positive, by summing the polarity of each word in the comment to give overall sentiment.

Before classification could be applied, certain preprocessing steps had to be completed. As YouTube comments are varied with lot of words and characters in the comments that were either unreadable by TextBlob or gave no indication of sentiment. To reduce error in the model, the redundant parts of comments had to be removed. We first used the emoji package in python to translate all emojis into a text description of the image. For example:

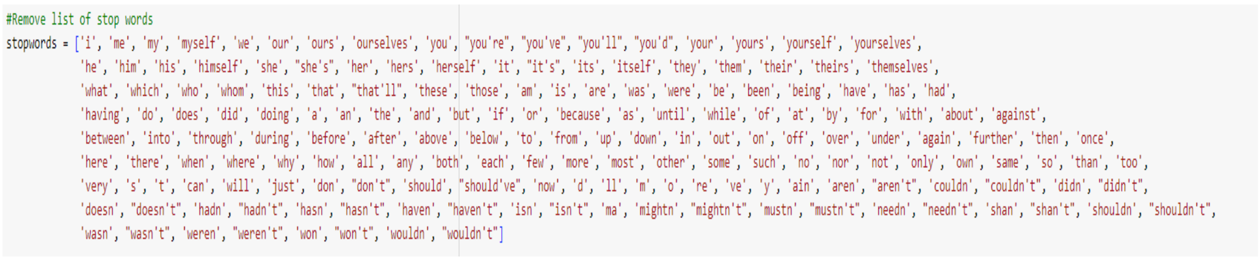


ANOTHER ONE 🔥🔥🔥🔥

This was one comment on the song "Lovin' on me", which translated into:
This gave TextBlob a better chance at correctly assigning sentiment, as a happy face emoji would translate to happy face, 'happy' would then be given positive polarity, thus making that

ANOTHER ONE firefirefirefire

comment deemed positive. The next pre-processing step was to remove irrelevant information from the comments. Many comments we found were user tagged through a username or email, this obviously gives no relation to sentiment, thus all punctuation and numbers were removed from all the comments. As well as punctuation, a list of stopwords were compiled, these were also words that gave no relation to polarity and could be removed. Figure 1 shows the stop words we used.



```
#Remove list of stop words
stopwords = ['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves',
             'he', 'him', 'his', 'himself', 'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves',
             'what', 'which', 'who', 'whom', 'this', 'that', "that'll", 'these', 'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had',
             'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against',
             'between', 'into', 'through', 'during', 'before', 'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under', 'again', 'further', 'then', 'once',
             'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both', 'each', 'few', 'more', 'most', 'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than', 'too',
             'very', 's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should've", 'now', 'd', 'll', 'm', 'o', 're', 've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "didn't",
             'doesn', "doesn't", 'hadn', "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn', "isn't", 'ma', 'mightn', "mightn't", 'mustn', "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shouldn't",
             'wasn', "wasn't", 'weren', "weren't", 'won', "won't", 'wouldn', "wouldn't"]
```

Figure 2: The list of relevant Stop Words

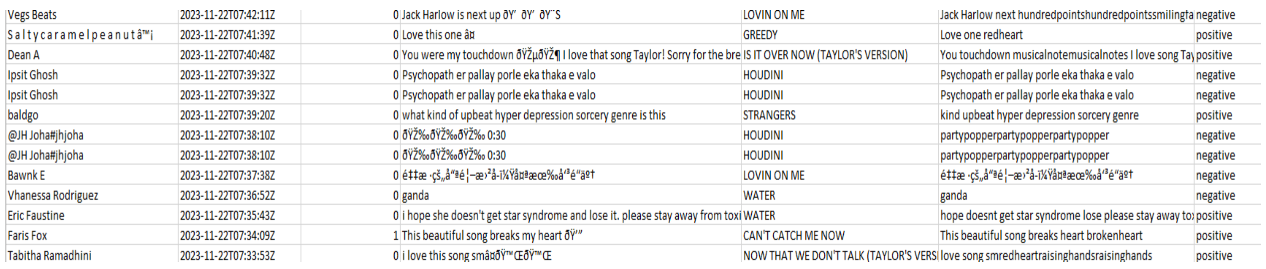
Finally, we applied a lemmatizer to generalise the meaning of words. This meant taking words without any nuance, to make the words as simple as possible. For example, the words 'great', 'better' and 'best' would all simply translate to the word 'good'. This simplifies the comments allowing improve probability of correct classification.

Ultimately the justification for these preprocessing steps is to simplify the comments into the least number of words, whilst keeping the best description of sentiment. This simplification is necessary as TextBlob uses a tokenization process where comments are broken down into individual words or tokens, giving each token in the string a separate polarity score, then averaging to gain the overall polarity. This means that reducing the number of tokens into the minimum will give minimal random error in our model.

After tokenization was achieved and the comments were classified, the processed comments and classifications were added to the dataset. Figure 3 Shows an example of some processed comments.

Classification

From this classification we found that out of the 512377 comments, 43076 were classified as negative, 163267 were classified positive and 306034 were classified neutral. As 59.7% of all classifications were deemed neutral, we can expect that this class has been over prescribed by TextBlob as it has struggled to gain a definitive polarity for a lot of comments thus chosen neutral. This however does not help in our goal in ranking the top 40, so we have decided to train machine learning models, to learn the difference between the negative and positive comments, to then properly classify all the neutrals. We will then use these models to stream



Vegs Beats	2023-11-22T07:42:11Z		0	Jack Harlow is next up ðŸ' ðŸ' ðŸ¨S	LOVIN ON ME	Jack Harlow next hundredpointshundredpointssmilingfa	negative
S a l t y c a r a m e l p e a n u t â™¡	2023-11-22T07:41:39Z		0	Love this one â¤	GREEDY	Love one redheart	positive
Dean A	2023-11-22T07:40:48Z		0	You were my touchdown ðŸŽµðŸŽ¶ I love that song Taylor! Sorry for the bre	IS IT OVER NOW (TAYLOR'S VERSION)	You touchdown musicalnotemusicalnotes I love song Tay	positive
Ipsit Ghosh	2023-11-22T07:39:32Z		0	Psychopath er pallay porle eka thaka e valo	HOUDINI	Psychopath er pallay porle eka thaka e valo	negative
Ipsit Ghosh	2023-11-22T07:39:32Z		0	Psychopath er pallay porle eka thaka e valo	HOUDINI	Psychopath er pallay porle eka thaka e valo	negative
baldgo	2023-11-22T07:39:20Z		0	what kind of upbeat hyper depression sorcery genre is this	STRANGERS	kind upbeat hyper depression sorcery genre	positive
@JH Joha#jhjoha	2023-11-22T07:38:10Z		0	ðŸŽ‰ðŸŽ‰ðŸŽ‰ 0:30	HOUDINI	partypopperpartypopperpartypopper	negative
@JH Joha#jhjoha	2023-11-22T07:38:10Z		0	ðŸŽ‰ðŸŽ‰ðŸŽ‰ 0:30	HOUDINI	partypopperpartypopperpartypopper	negative
Bawnk E	2023-11-22T07:37:38Z		0	é‡‡æ ·çš„å"ªé¦–æ›²å-ï¼Ÿå¤ªæœ‰å³é"äº†	LOVIN ON ME	é‡‡æ ·çš„å"ªé¦–æ›²å-ï¼Ÿå¤ªæœ‰å³é"äº†	negative
Vhanessa Rodriguez	2023-11-22T07:36:52Z		0	ganda	WATER	ganda	negative
Eric Faustine	2023-11-22T07:35:43Z		0	i hope she doesn't get star syndrome and lose it. please stay away from toxi	WATER	hope doesnt get star syndrome lose please stay away to:	positive
Faris Fox	2023-11-22T07:34:09Z		1	This beautiful song breaks my heart ðŸ'"	CAN'T CATCH ME NOW	This beautiful song breaks heart brokenheart	positive
Tabitha Ramadhini	2023-11-22T07:33:53Z		0	i love this song smâ¤ðŸ™Œ ðŸ™Œ	NOW THAT WE DON'T TALK (TAYLOR'S VERSI	love song smredheartraisinghandsraisinghands	positive

Figure 3: Examples of Post Processed data

and classify comments in real time.

To measure the accuracy of our models and determine which one is the best at classifying comments, we will be comparing the area under the curve of a ROC curve given by each model. A ROC curve shows the true positive rate (the amount of correct positive predictions) on the y-axis, by the false positive rate (the amount of incorrect positive predictions) on the x-axis[5]. This would mean that the objective is to gain an AUC score of exactly 1, a 100 success rate, however this is rarely feasible, and an extremely good score would be above 90%. A poor score would be just above 0.5, as if we have a ratio of 0.5 that means that half the predictions were wrong, and the model is effectively a random guess. A score below 0.5 is very rare and would mean the model is worse than a random guess.

To train the model we had to use the tensorflow package for python, from this we padded and sequenced the polarity scores, with a max length of 500. This made all the input data be the same length and form, making it easily applied to the sklearn packages to fit the model. Using the positive and negative classification as the basis, we split the data into a $70 - 30\%$ split, training and test. This is necessary because we need as much data as possible in the training set to teach the model how to classify properly, however we still need enough data in the test set to make sure the predictions are correct. After the splitting we scaled the data to be ranging from 0 to 1 through the equation:

$$\text{Scaled Data} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

This allows the values to be inputted into the models effectively.

The first model we implemented was a logistic regression. This simple machine learning model that uses weighted coefficient features to binary classify objects. A decision boundary is constructed where points that lie on either side of the boundary are separated into different classes. Through the equation:

$$Cost(y, \bar{y}) = -ylog(\bar{y}) - (1 - y)log(1 - \bar{y})$$

The cost is described which is equivalent to the AUC of the ROC curve, and describes the amount of correct to wrong predictions. This can be improved through the use of the gradient descent method and tuned hyperparameters; however, we did not have the time to implement these methods so the default parameters were used. From this model we achieved an AUC score of 0.622, this shows that the classification model is not very accurate, only correctly classifying 62.2% of comments correctly. As this was such a low score, we did not want to waste time tuning the model, and just decided to try a new model to see if it is more accurate.

The next model we tried was a simple decision tree. The fundamental idea behind the decision tree is to recursively split the dataset into subsets based on a significant attribute. In our case, we will split the dataset depending on the polarity score, giving a threshold were if it is greater than the threshold it will be deemed positive. This splitting is completed recursively, at each node recalculating the feature selection until all conditions are met. As we are completing binary classification, criteria parameter used was a 'gini' impurity, this tells the model to binary classify instead of complete regression. The max leaf nodes used was 100 and the minimum was 1, leaving the final parameter of max_depth to be 100. From these parameters we gained a AUC score of 0.729 or 72.9%, showing an improvement over the logistic regression.

Due to this improvement, we decided to carry on with this family of machine learning algorithms. As 1 decision tree gave a respectable AUC score, we decided to fit a Random Forest. Random Forests uses a multitude of decision trees with randomly selected starting nodes, then averaging the end classification find the overall classification. [6]This method should increase the AUC score further as using multiple decision trees will always give a greater score, at the cost of computational efficiency. The hyperparameters used in this case were n_estmators = 100 and random_state = 42. These parameters are not the most efficient, as again, we did not have enough time to properly parameter tune the model. This model gave an overall AUC score of 0.817 meaning 81.7% of classification were correct, making the Random Forest substantially the best model, as expected.

Figure 4 shows the output of the ROC curve:

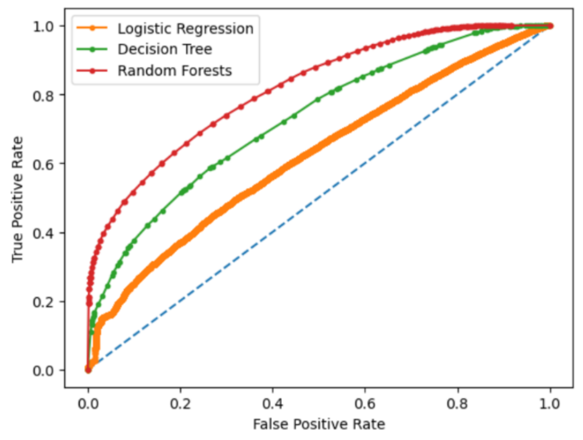


Figure 4: A ROC Curve displaying results for Logistic Regression, Decision tree and Random Forests.

When comparing the total counts of positive and negative classifications we see a lot of variation between models. Out of the 500,000 comments grouped, the Logistic Regression model thought 52.1% of the comments were positive, whereas the Decision Tree and Random Forest thought 37.9% and 57.0% respectively. This adds to the evidence of the Random Forests being the most accurate model as we would naturally expect there to be more positive comments then negative. As we are comparing the comments of music videos, we would assume that the majority of comments are going to be written by fans of the artist, thus would be more likely to be positive in sentiment.

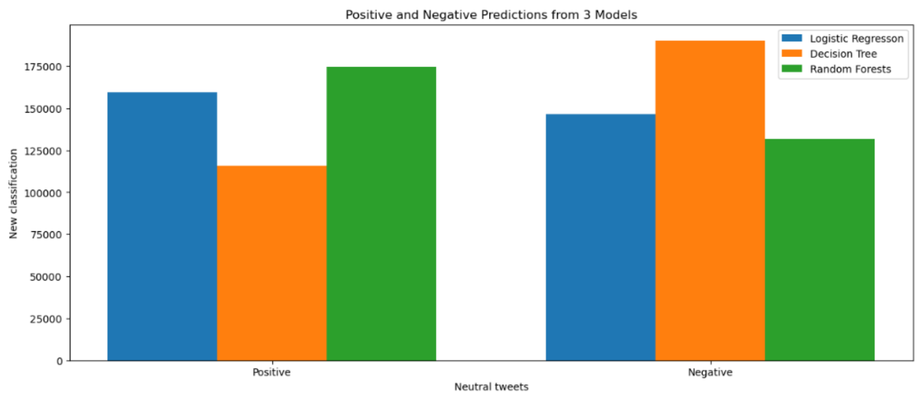


Figure 5: Total Number of Positive and Negative Classifications per Model.

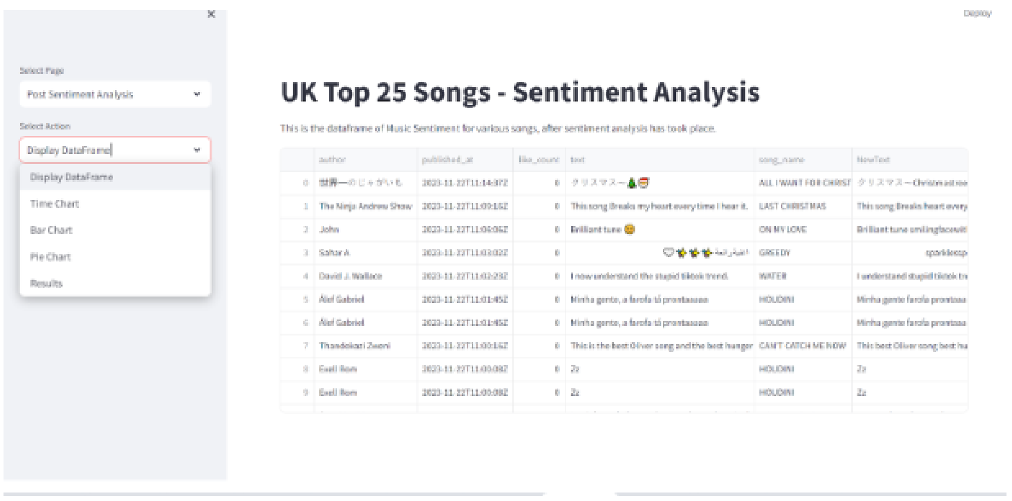
Creating a Dashboard

One of the most important aims of our project was in creating the interactive dashboard. Often, dashboards are made with commonly used softwares, for example in Tableau[8] or Microsoft PowerBI [7], however without access to these softwares, it initially proved quite difficult. Hence, to combat this, we used a python package called streamlit. Streamlit is an open-source package which allows for the creation of dashboards with only python code, without the need for front-end languages such as html. [9]

Within this part of the project, we had three key aims for the development of the dashboard. The first was to provide a wide range of visualisations, which allow for users to understand the results easily. Secondly, we wished to display the results of the batch data, which can give understanding of the initial sentiment analysis. Finally, we wished to be able to use the dashboard in real time, with updates being made for new data.

Batch Data Dashboard

Figure 5 shows what the dashboard looks like upon loading. It is laid out in a simple, easy to use way, with a navigation bar on the left hand side. It allows for the user to navigate between the batch data analysis, and the new data for real time analysis. The first page simply displays the data post sentiment analysis for the user to use. We include this to allow new users to gain understanding into what the data is.



UK Top 25 Songs - Sentiment Analysis

This is the dataframe of Music Sentiment for various songs, after sentiment analysis has took place.

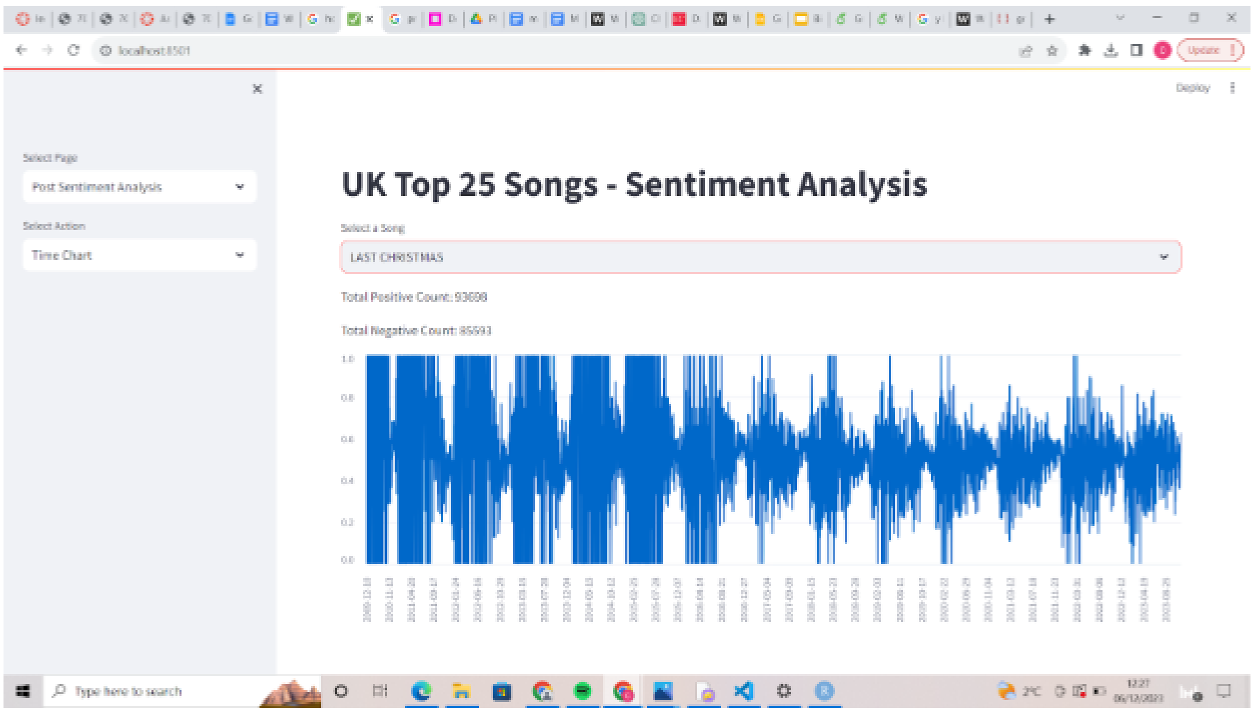
	author	published_at	like_count	text	song_name	NewText
0	世界一のじゃがいも	2023-11-22T11:14:37Z	0	クリスマス一🎄🎅	ALL I WANT FOR CHRIST	クリスマス一Christmastree
1	The Ninja Andrew Shaw	2023-11-22T11:09:15Z	0	This song Breaks my heart every time I hear it.	LAST CHRISTMAS	This song Breaks heart every
2	John	2023-11-22T11:06:06Z	0	Brilliant tune 😊	ON MY LOVE	Brilliant tune smilingfacewit
3	Sahar A	2023-11-22T11:03:02Z	0	🤍✨✨✨ [illegible]	GREEDY	sparkles
4	David J. Wallace	2023-11-22T11:02:23Z	0	I now understand the stupid tiktok trend.	WATER	I understand stupid tiktok tr
5	Alef Gabriel	2023-11-22T11:01:45Z	0	Minha gente, a farofa tá prontaaaaa	HOUDINI	Minha gente farofa prontaaa
6	Alef Gabriel	2023-11-22T11:01:45Z	0	Minha gente, a farofa tá prontaaaaa	HOUDINI	Minha gente farofa prontaaa
7	Thandokazi Zwoni	2023-11-22T11:00:16Z	0	This is the best Oliver song and the best hunger	CAN'T CATCH ME NOW	This best Oliver song best hu
8	Exell Rom	2023-11-22T11:00:08Z	0	Zz	HOUDINI	Zz
9	Exell Rom	2023-11-22T11:00:08Z	0	Zz	HOUDINI	Zz

Figure 6: Dashboard Homepage

Following this, we included numerous pages with various charts. We put each chart on its separate page to keep the layout simple and easy to understand. The first chart we included was a time chart. This used a drop down selection bar, to select a song, and displayed the proportion of positive comments over time. This was achieved by summing the number of positive comments on a given day, over the total comments on that day. This worked well for songs with a large number of comments, however occasionally had issues with songs with fewer comments. Figure 6 shows an example for the song ”Last Christmas”. We see this is a particularly interesting example, with what appears to be a yearly cycle. Contextually, this makes sense as people listen to Christmas music seasonally, and huge the spikes we see appear mid year, when there is fewer comments and more easily influenced.
The second chart was a bar chart. This displays the proportion of positive comments to the proportion of negative comments, with a selection to display either all the songs, the positive songs or the negative songs. This is a way to gain quick insight into which songs are positively viewed. Figure 7 shows both the positive and negative songs. We see that there is a lot more positively perceived songs than their are negatively perceived, and of those who are negatively perceived they appear a lot closer to a fifty fifty split. Contextually, this makes sense that the most streamed songs currently have generally positive comments.

The third chart is a simple pie chart with a selection bar for which song to view. These pie charts also display the proportion of positive to negative comments. However, we included two bar charts, one with the likes included and one without. We decided to include the likes as a metric for an additional comment, with for example 100 likes being 100 comments. Whilst this is not entirely fair, as the same person can like multiple comments much easier than leaving multiple comments, we decided since people can be equally likely to like negative or positive comments, it is a fair way to include them. As we see, when we compare the two charts for all the comments, it takes the comments from a near fifty fifty split, to a proportion of 39.2% negative to 60.8% positive.

Finally, we included the results page. The initial objective of our project was to provide an



Select Page

Post Sentiment Analysis

Select Action

Time Chart

UK Top 25 Songs - Sentiment Analysis

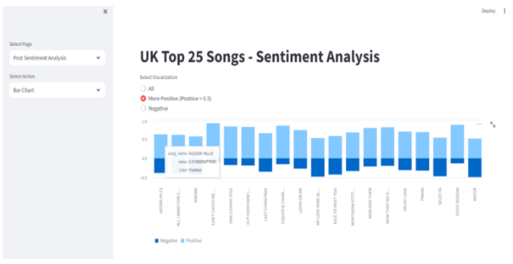
Select a Song

LAST CHRISTMAS

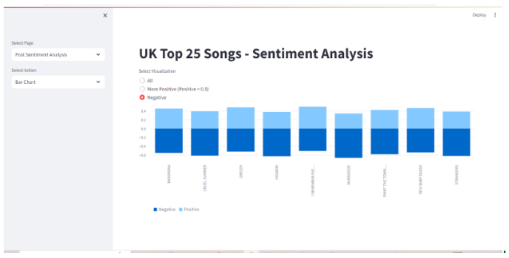
Total Positive Count: 93698

Total Negative Count: 85593

Figure 7: A Line Graph for Last Christmas

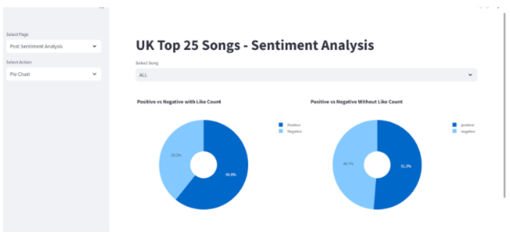


(a) Positively Perceived Songs

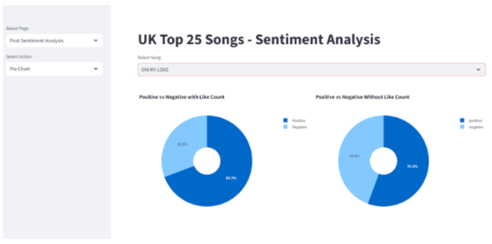


(b) Negatively Perceived Songs

Figure 8: Bar Charts of Proportions Positive Negative



(a) All Songs

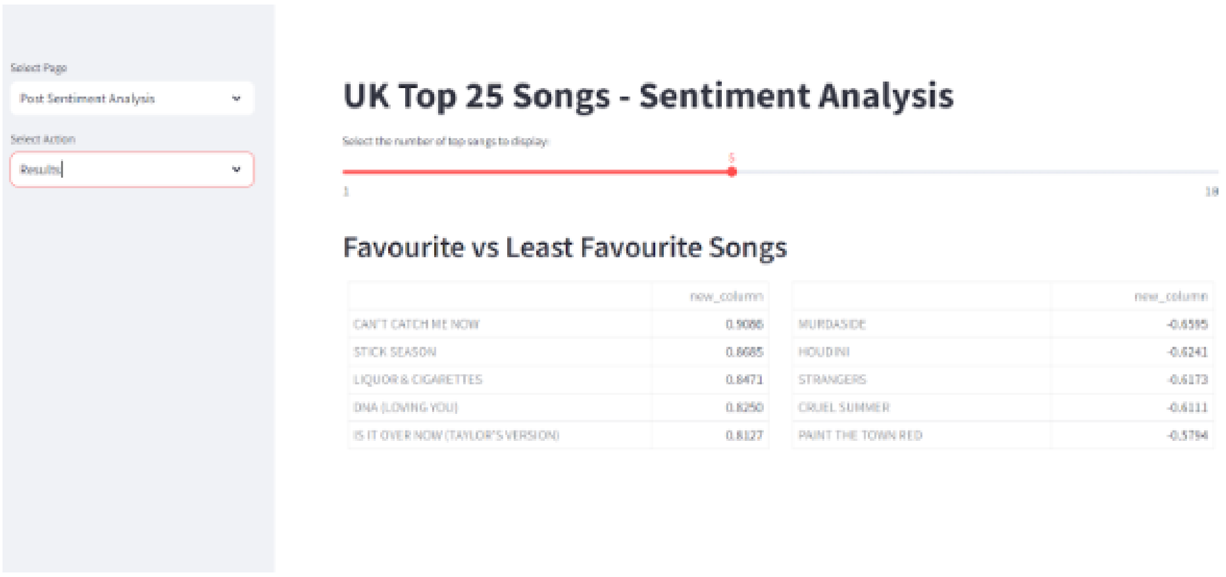


(b) Song - On My Love

Figure 9: Pie Charts of Proportions Positive to Negative With and Without like count.

objective way of ranking the songs, based not on the number of streams but how positively the song is perceived. We can do this by studying the discrepancies between our sentiment ranking of the songs against the actual top 40 ranking. The top 5 most well received songs were given as ‘Can’t catch me now’, ‘stick season’, ‘liquor & cigarettes’, ‘DNA’ and ‘Is it over now’, with sentiment scores of 0.9086, 0.8685, 0.8471, 0.8250 and 0.8127 respectively. From this we would expect these songs to be at the very top of the top 40 list, their actual rankings however were much more spread out. ‘Can’t catch me now’ the most positive song, was ranked 25th on the top 40, ‘Liquor and cigarettes’ ranked 17th and ‘DNA’ ranked 14th, the discrepancies were less severe in the other songs with ‘stick season’ ranking 5th in the top 40 and ‘is it over now’ ranking 7th. As all 5 of the positive songs were in the top half of the of the top 40 ranking, we can determine a small correlation between the positive ranking of our models and the ranking in the top 40.

On the negative side, we would expect these songs to be at the bottom of the top 40 ranking. Unlike on the positive side, the discrepancies in this section were very large suggesting that counting the proportion of negative comments does not suggest anything about the position at which the song will place on the top 40. For example the second most negative comment section was associated with the song ‘Houdini’ however this placed 2nd on the top 40, ‘Cruel summer’ was another good example, ranking 4th on our negative list and 9th in the top 40. This large discrepancy between our model and the real ranking of the songs could be explained by the overall lower negative scores then the positive ones. In the top 5 positive songs the average score was given as 0.85238 meaning that 85.2% of all comments were positive whereas in the negatives, the average score was -0.61828, meaning 61.8% of comments were negative. As the negative score is closer to 0, the proportion of positive to negative was less than in the highly positive songs, meaning there was a larger blend and positives and negatives in the highly negative comments. As the comments are less distinctly negative we would expect to measure more variation in our rankings and the top 40 ranking, as the difference between songs is reduced.



UK Top 25 Songs - Sentiment Analysis

Favourite vs Least Favourite Songs

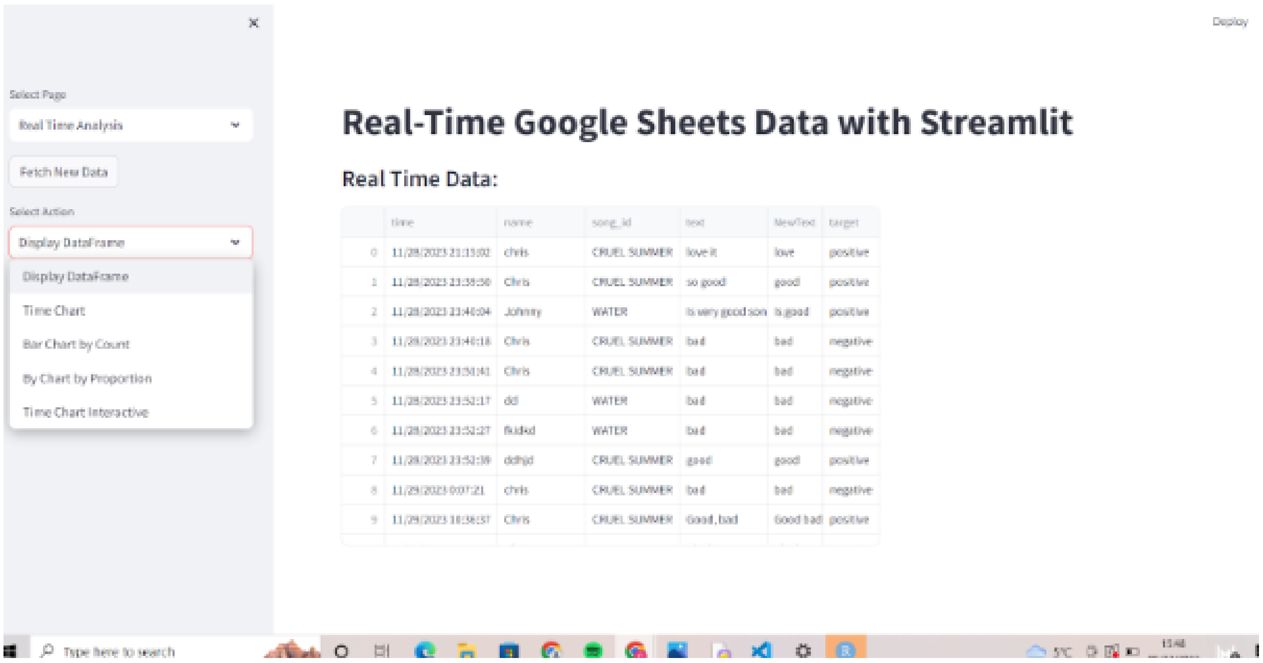
	new_column
CAN'T CATCH ME NOW	0.9086
STICK SEASON	0.8685
LIQUOR & CIGARETTES	0.8471
DNA (LOVING YOU)	0.8250
IS IT OVER NOW (TAYLOR'S VERSION)	0.8127

	new_column
MURDASIDE	-0.6595
HOUDINI	-0.6241
STRANGERS	-0.6173
CRUEL SUMMER	-0.6111
PAINT THE TOWN RED	-0.5794

Figure 10: The Results Page for Batch Data

Real Time Data Dashboard

One way we decided to obtain real time source data was from others in our cohort, during our presentation. We did this by creating a google form, which connected to a spreadsheet which we called to update via a fetch button. One important thing to note is that whilst this is not the truest form of real time social media analysis, it ensures the comments were relevant and could be easily changed for another method of collecting data. Whilst we could have done this by timer, we faced issues with the Streamlit package. With Streamlit, it runs the entire code each time something is called, hence if it was to refresh on a timer, it often spent huge amounts of time refreshing when not needed. With a button, this occurred only when we wished it to.



Real-Time Google Sheets Data with Streamlit

Real Time Data:

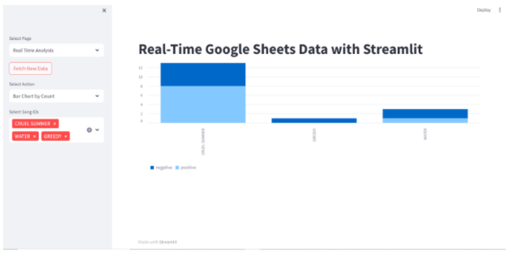
	time	name	song_id	text	NewText	target
0	11/28/2023 21:13:02	chris	CRUEL SUMMER	love it	love	positive
1	11/28/2023 23:39:50	Chris	CRUEL SUMMER	so good	good	positive
2	11/28/2023 23:40:04	Johnny	WATER	is very good son	is good	positive
3	11/28/2023 23:40:18	Chris	CRUEL SUMMER	bad	bad	negative
4	11/28/2023 23:51:41	Chris	CRUEL SUMMER	bad	bad	negative
5	11/28/2023 23:52:17	dd	WATER	bad	bad	negative
6	11/28/2023 23:52:27	fkdkd	WATER	bad	bad	negative
7	11/28/2023 23:52:39	ddhjd	CRUEL SUMMER	good	good	positive
8	11/29/2023 0:07:21	chris	CRUEL SUMMER	bad	bad	negative
9	11/29/2023 10:36:37	Chris	CRUEL SUMMER	Good, bad	Good bad	positive

Figure 11: Real Time Data Dashboard Homepage

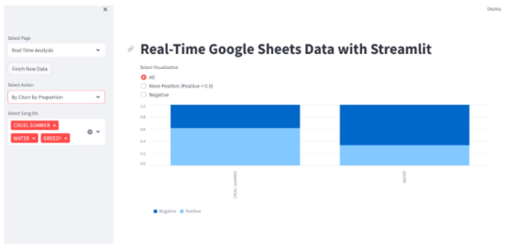
Figure 11 shows the homepage of the real time data, which contains the real time data received during our presentation. It follows the same structure as the batch data, to allow for easy application of our already created graphs, and for consistency in the layout. Similarly, the side bar looks almost exactly the same, only this time with the choose a song multi chose button, and fetch new data button. We include the first as since there was less data and less songs, it was easier to provide the choice in this manner.

Then, we applied mostly the same charts to the new data. However, this time I included a bar chart by count, as well as a bar chart by proportion. This was simply due to the difference in the scale of the amount of data we were using. Figure 12 shows these bar charts, as we see they do not look as impressive but that is due to the volume.

The only graph we include which was different was an interactive line graph. This had graph plots individual for positive and negative in each frame, and plays the graph. Similarly, we



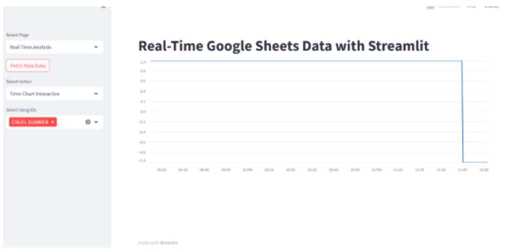
(a) Bar Chart by Count



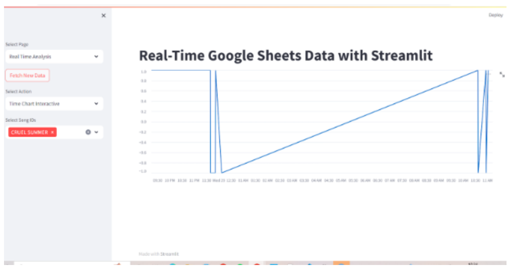
(b) Bar Chart by Proportion

Figure 12: Bar Charts in Real Time

can zoom into certain aspects. Initially, we wished for this to update automatically when there was new data, however this proved impossible due to the limits of Streamlit. Similarly, it only worked when viewing one song at a time, as with multiple songs there would be breaks between overlapping times, however it gave an interesting metric of how opinions of song may change in the real time.



(a) Time Graph for Early Time.

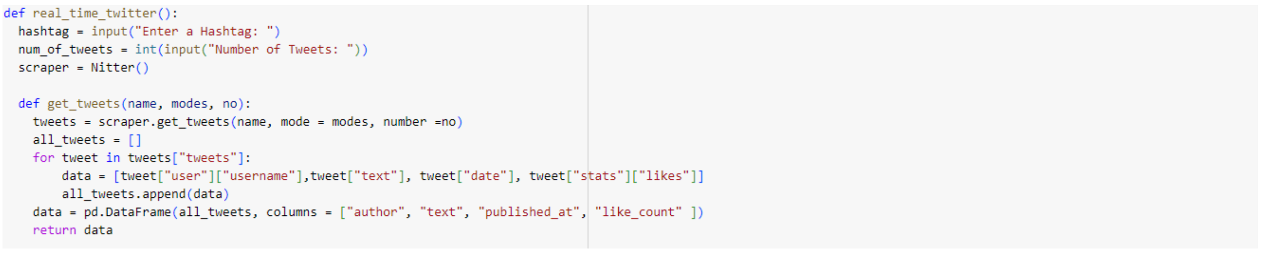


(b) Time Graph for Late Time.

Figure 13: Interactive Time Graphs in Real time.

Real Time Data - Twitter Function

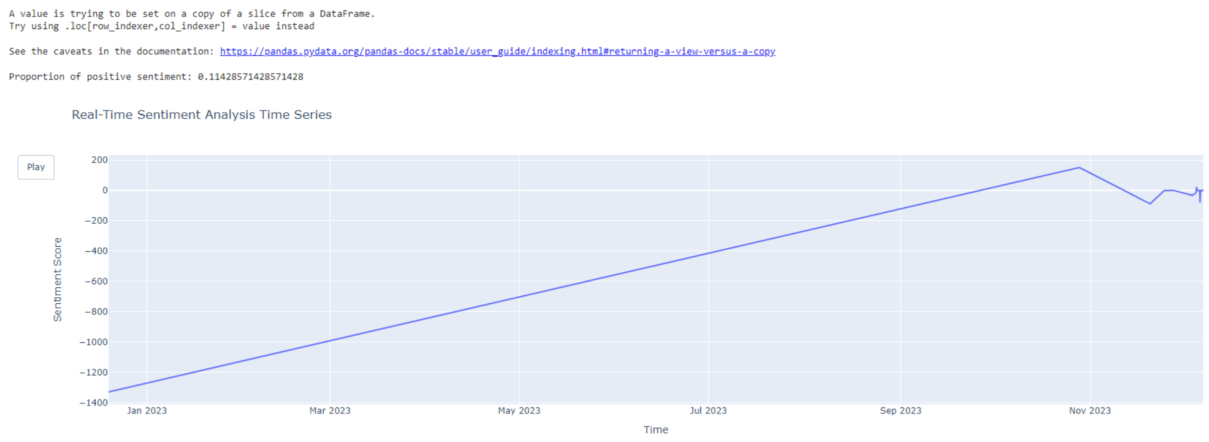
Whilst this method was a good way of seeing the real time opinion of those in the class, it is not quite representative of the public opinion, instead just a group of like minded people. To improve, we wished to obtain more new data. To do this, we decided to scrape tweets from Nitter. Nitter is an open-source X viewer, which allows for scraping, unlike X. [10]



```
def real_time_twitter():
  hashtag = input("Enter a Hashtag: ")
  num_of_tweets = int(input("Number of Tweets: "))
  scraper = Nitter()

  def get_tweets(name, modes, no):
    tweets = scraper.get_tweets(name, mode = modes, number =no)
    all_tweets = []
    for tweet in tweets["tweets"]:
        data = [tweet["user"]["username"],tweet["text"], tweet["date"], tweet["stats"]["likes"]]
        all_tweets.append(data)
    data = pd.DataFrame(all_tweets, columns = ["author", "text", "published_at", "like_count" ])
    return data
```

Figure 14: Scaping Function

This function takes an input for the hashtag of the tweet you wish to view and number of tweets you wish to obtain. From here, the get_tweet() function scrapes the tweets, and appends to a dataframe, which can be used for analysis. In the hidden part of real_time_twitter(), sentiment analysis is applied on this new data, and an interactive line plot is then plotted, along with printing the proportion positive of the tweets. One important thing to consider when using this function is picking hashtags which do not overlap. For example, the song "Water" would be a bad choice, with how common a hashtag that would be, and contain a huge amount of junk. We will use these functions to view two songs which have since the initial data collection, have since joined the charts, namely "jinglebells" and "runaway".

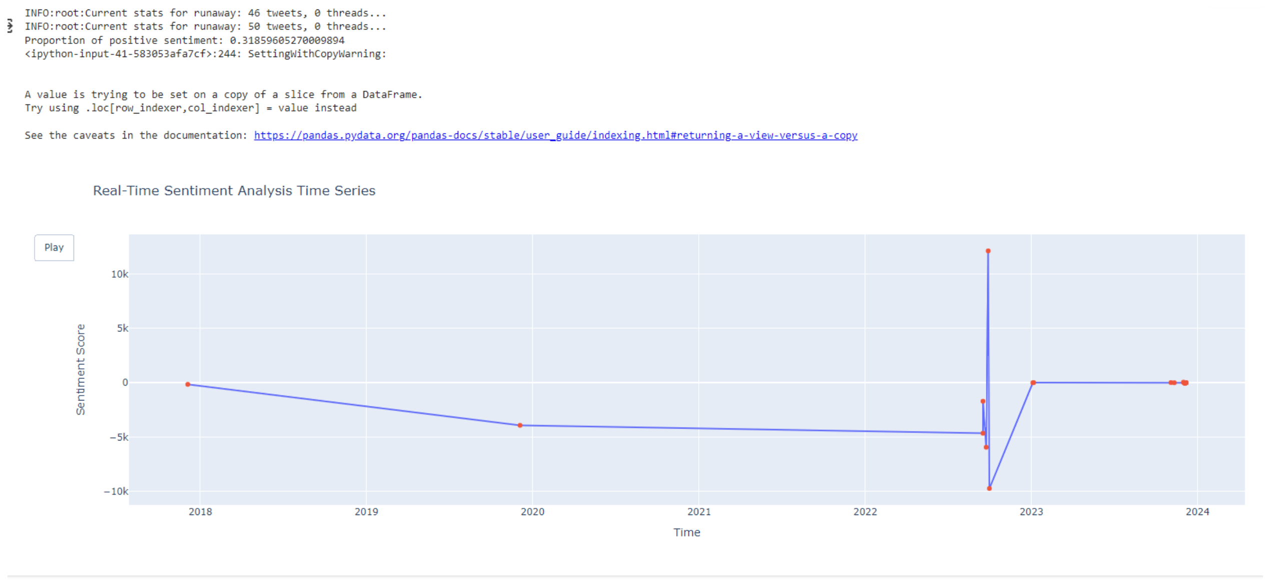


```
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

Proportion of positive sentiment: 0.11428571428571428
```

Figure 15: Graph of 50 Scraped Tweets with Hashtag "Jinglebells"

Figure 15 shows the use of this function for the hashtag "jinglebells". As we see, there was an extremely highly liked negative comment before January this year, following which no comments for the rest of the year until November. This is jinglebells is a christmas song. From here, There was a range of values around 0, however mostly negative tweets. One thing to note is the scale of the graph is not perfect, however I could not apply a logarithmic transformation due to the negative values. We see the proportion positive is 0.11. Whilst this might be massively influenced by one or two tweets due to the inclusion of "likes" as a scalar, this shows that people generally do not enjoy this song.One thing to note is this is significantly lower than any of the negatives obtained from Youtube data. Whilst this does have a lot less tweets and comments also, it may show that people are more inclined to leave negative comments on twitter than they are on Youtube.

Another example is shown in Figure 16. Runaway is a song by the controversial artist Kanye West, which despite being released in 2015, releasely re-entered the charts after going viral on platform Tiktok. We see that the range of the dates is large, with very few tweets between 2018 and 2022, with a spike in September 2022, and then a recent collection of tweets. In 2022, Kanye West controversy began, with various events such as his divorce, and reports of antisemitism[11]. This would explain why tweets during this time were so polarised, with huge numbers of likes. Then, following this, the more recent spike is explained by the recent charting, however since he is not in the media as much now, it makes sense the tweets would



```
INFO:root:Current stats for runaway: 46 tweets, 0 threads...
INFO:root:Current stats for runaway: 50 tweets, 0 threads...
Proportion of positive sentiment: 0.31859605270009894
<ipython-input-41-583053afa7cf>:244: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
```

Figure 16: Graph of 50 Scraped Tweets with Hashtag "Runaway"

not be liked as much. Similarly, we see the proportion positive is 0.32.

Conclusion

To conclude, we have successfully created a way to compare the rankings of songs in the UK top 40, to the public opinion and seen there is no obvious correlation. Furthermore, we improved our dashboard to allow it to be applied to new data about new songs and applied further classification to new data from multiple sources. We classified the comments using three machine learning models, Logistic Regression, with an AUC score of 0.622, a decision tree with 0.729 and Random Forest with 0.817. These could have been further improved with some hyperparameter tuning, however with time constraints that wasn't feasible. It is important to note it could similarly be applied to data about things other than music, any string character could be inputted through the preprocessing and the classifications could be applied.

References

[1] analyticswithadam. (2023) *YouTube Comments Advanced.* Created on October 28. Available at: `https://github.com/analyticswithadam/Python/blob/main/YouTube_Comments_Advanced.ipynb` (Accessed: 08 December 2023).

[2] bocchilorenzo. (n.d.) *ntscraper.* Available at: `https://github.com/bocchilorenzo/ntscraper` (Accessed: 08 December 2023).

[3] Official Charts. (n.d.) *Official Singles Chart top 40.* Available at: `https://www.officialcharts.com/charts/uk-top-40-singles-chart/` (Accessed: 08 December 2023).

[4] Loria, S. (2023) *TextBlob: Simplified Text Processing*, Version 0.16.0. Available at: `https://textblob.readthedocs.io/` (Accessed: 04 December 2023).

[5] Google. (n.d.) *Classification: Roc curve and AUC — machine learning — Google for Developers.* Available at: `https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc` (Accessed: 08 December 2023).

[6] IBM. (n.d.) *What is Random Forest?.* Available at: `https://www.ibm.com/topics/random-forest` (Accessed: 04 December 2023).

[7] Microsoft. (n.d.) *Microsoft Power BI.* Available at: `https://powerbi.microsoft.com/en-gb/` (Accessed: 06 December 2023).

[8] Tableau Software. (n.d.) *Tableau.* Available at: `https://www.tableau.com/` (Accessed: 06 December 2023).

[9] Streamlit, Inc. (n.d.) *Streamlit.* Available at: `https://streamlit.io/` (Accessed: 08 December 2023).

[10] Nitter Project. (n.d.) *Nitter GitHub Repository.* Available at: `https://github.com/zedeus/nitter` (Accessed: 02 December 2023).

[11] Knibbs, J. (2022) Kanye West's 2022 Timeline of Controversy Including Hitler and Swastika References, *Evening Standard.* Available at: `https://www.standard.co.uk/news/uk/kanye-west-ye-2022-twitter-controversy-b1045264.html` (Accessed: 05 December 2023).